5B/6B-T, 3B/4B-T AND PARTITIONED 8B/10B-T AND 10B/12B TRANSMISSION CODES, AND THEIR IMPLEMENTATION FOR HIGH OPERATING RATES

Field of the Invention

The present invention relates to transmission codes and, more particularly, relates to methods and apparatuses used to produce and interpret 5B/6B, 3B/4B, 8B/10B, and 10B/12B transmission codes.

Background of the Invention

In a partitioned 8B/10B transmission code, an input vector having eight bits is partitioned into two smaller vectors having three and five bits, respectively. Coded vectors having four and six bits, respectively, are created from the partitioned vectors through use of 3B/4B and 5B/6B transmission code vector sets. The resultant coded vectors then form a single ten-bit coded vector suitable for transmission. Generally, a control input generates control characters readily recognized as other than the 256 data characters in an 8B/10B transmission code.

The original partitioned 8B/10B transmission code, introduced more than 20 years ago, was designed to minimize the number of gates required for encoding and decoding. The original partitioned 8B/10B transmission code is described in Franaszek and Widmer, U.S. Patent No. 4,486,739, issued December 4, 1984, the disclosure of which is hereby incorporated by reference. The original circuitry was implemented with emitter coupled logic, some versions were also done in bipolar transistor/transistor logic, and this was followed by several complementary metal oxide silicon designs. One example is PCT No. US02/13798, entitled "8B/10B Encoding and Decoding for High Speed Applications," claiming the benefit of U.S. Patent Application No. 60/289,556 that was filed on May 8, 2001, the disclosure of which is hereby incorporated by reference.

Lately, design efforts have concentrated on high operating rates. Traditional means for achieving higher operating rates for transmission codes, such as the partitioned 8B/10B transmission code, have involved the parallel operation of several encoders and decoders. Current important potential applications for the partitioned 8B/10B transmission code and its 5B/6B component are for very wide high speed buses using sets of parallel serial links,

with each serial link operating up to dozens of Gbaud for short links which require short latencies for performance reasons. Operation with a single CoDec (coder/decoder) circuit for each serial link, or a reduction of the multiplexing ratios required at both ends to accommodate parallel CoDec circuits required to serve a single link, is desirable to improve the latency aspect.

Although conventional 8B/10B encoding and decoding work well for a large number of applications, the conventional codes could be improved, particularly in operating rates and latency. Thus, what is needed is a partitioned 8B/10B transmission code and apparatus using the same that allow high operating rates and low latency.

Additionally, some applications are compatible with 5-bit data units. It would be beneficial to enable the use of 5B/6B transmission codes with such applications in the form, for instance, of 10B/12B transmission codes. Consequently, improvements to 10B/12B transmission codes are desired.

Summary of the Invention

The present invention provides techniques for implementing 5B/6B, 3B/4B and partitioned 8B/10B and 10B/12B transmission codes for high operating rates.

In an exemplary aspect of the invention, techniques are disclosed for translating five-bit source vectors, each having five source bits, from a number of five-bit source vectors into six-bit coded vectors. A sixth bit having a default value is appended to the source vectors. Selected one to three individual source bits are complemented for a minority of the source vectors. The coded vectors are disparity independent with a single representation or disparity dependent with a primary and an alternate representation, where the alternate representation is a complement of the primary representation.

In another exemplary aspect of the invention, techniques are disclosed for translating three-bit source vectors, together with one or more control inputs, into nine four-bit coded vectors. The source vectors have three source bits. A fourth bit having a default value is appended to the source vectors. A single individual source bit is complemented for a minority of the source vectors. The coded vectors are disparity independent with a single representation or disparity dependent with a primary and an alternate representation, where the alternate representation is a complement of the primary representation.

In another exemplary aspect of the invention, techniques are presented for encoding a partitioned 10B/12B transmission code. Pairs of five-bit source vectors are operated on to produce pairs of six-bit coded vectors. A starting disparity is determined. A synchronizing coded pattern is generated based on the starting disparity. When the starting disparity is positive, a predetermined pattern is generated. When the starting disparity is negative, a complement of the predetermined pattern is generated.

A more complete understanding of the present invention, as well as further features and advantages of the present invention, will be obtained by reference to the following detailed description and drawings.

Brief Description of the Drawings

FIG. 1A is a trellis diagram illustrating a number of primary vectors of a 5B/6B-T code portion of an 8B/10B-T transmission code according to an embodiment of the invention;

FIG. 1B is a trellis diagram illustrating a number of primary vectors of a 5B/6B-T code portion of an 8B/10B-T transmission code according to an embodiment of the invention;

FIG. 2 is a trellis diagram illustrating a primary vector of a 5B/6B-T code portion of an 8B/10B-T transmission code according to an embodiment of the invention;

FIG. 3 is a trellis diagram illustrating a number of primary vectors of a 5B/6B-T code portion of an 8B/10B-T transmission code according to an embodiment of the invention;

FIG. 4 is a trellis diagram illustrating a number of primary vectors of a 5B/6B-T code portion of an 8B/10B-T transmission code according to an embodiment of the invention;

FIG. 5 is a coding table for the 5B/6B-T code portion of an 8B/10B-T transmission code according to an embodiment of the invention;

FIG. 6 is a coding table for the 3B/4B-T code portion of an 8B/10B-T transmission code according to an embodiment of the invention;

FIG. 7 is a coding table for a basic set of control characters for an 8B/10B-T transmission code according to an embodiment of the invention;

FIG. 8 is a coding table for an additional set of control characters for an 8B/10B-T transmission code according to an embodiment of the invention;

FIG. 9A is a circuit diagram of the bit encoding portion of an 8B/10B-T encoder according to an embodiment of the invention;

FIG. 9B is a circuit diagram of the disparity control portion of an 8B/10B-T encoder according to an embodiment of the invention;

FIG. 10 illustrates exemplary circuitry using the 8B/10B-T encoder shown in FIGS. 9A and 9B to compute disparity for a single byte through a faster implementation;

FIG. 11 illustrates exemplary circuitry using the 8B/10B-T encoder shown in FIGS. 9A and 9B to compute disparity for a single byte through a slower implementation;

FIG. 12 illustrates exemplary circuitry using the encoder shown in FIGS. 9A and 9B to compute disparity for four bytes through a faster implementation;

FIG. 13 illustrates exemplary circuitry using the encoder shown in FIGS. 9A and 9B to compute disparity for four bytes through a slower implementation;

FIG. 14 is a decoding table for decoding the 6B/5B-T code portion of a 10B/8B-T transmission code according to an embodiment of the invention;

FIG. 15A is a circuit diagram of the decoding portion of a 6B/5B-T decoder according to an embodiment of the invention;

FIG. 15B is a circuit diagram of the disparity and error checking portion of a 6B/5B-T decoder according to an embodiment of the invention;

FIG. 16 is a decoding table for decoding the 4B/3B-T code portion of a 10B/8B-T transmission code according to an embodiment of the invention;

FIG. 17 is a circuit diagram of a 4B/3B-T decoder and error checks according to an embodiment of the invention;

FIG. 18 is a circuit diagram of a 10B/8B-T decoder using the 6B/5B-T decoder of FIGS. 15A and 15B and the 4B/3B-T decoder of FIG. 17;

FIGS. 19 and 20 illustrate exemplary circuitry using the 8B/10B-T decoder shown in FIG. 18 to compute disparity for a single byte through faster and slower, respectively, implementations;

FIG. 21 is a four-byte 10B/8B-T decoder;

FIG. 22 is a trellis diagram of a 10B/12B-T transmission code;

FIG. 23 is a table showing 5B/6B-T encoding for 10B/12B-T control characters;

FIG. 24 is a table of 6B trailers of the K3 vector used to form 12B control characters;

FIG. 25A is a circuit diagram for the 5B/6B-T portion of the bit encoding of a 10B/12B encoder;

FIG. 25B is a circuit diagram of the disparity control portion of the 10B/12B encoder;

FIG. 26A is a circuit diagram of the 6B/5B-T decoding portion of the 12B/10B decoding circuit; and

FIG. 26B is a circuit diagram of the disparity and error checking portion of the 12B/10B decoding circuit.

Detailed Description of Preferred Embodiments

The present invention provides techniques for speeding up encoding and decoding for 8B/10B and 10B/12B transmission codes.

A. Introduction

The appendage "-T" (i.e., "Turbo") is added to the references to the transmission codes used herein in order to distinguish these codes from conventional codes. The new 8B/10B-T transmission code retains the 5B/6B-T and the 3B/4B-T partitions. The codes presented herein are designed for high speed operation. Many of the changes between the conventional transmission codes and the transmission codes described herein are in the 5B/6B-T domain. For instance, for both the 5B/6B-T encoding and decoding described herein, fewer modifications of bit positions, in fewer vectors, are performed as compared to conventional 5B/6B techniques. As another example, the S-Function, which has an important purpose of preventing false commas, has been reduced to the minimum required to maintain the singularity of the comma at the expense of more frequent single runs of five in random data. A comma generally indicates proper byte boundaries and can be used for instantaneous acquisition or verification of byte synchronization. The K28.7 comma character of the traditional code has been swapped with a formerly invalid control character K3.7 ('1100001110' and its complement) which is not a comma character but has no sequence restrictions. Nine additional control characters have been defined and are listed in the table shown in FIG. 8.

Notation

The signal names used in the equations of this document do not reflect any logic levels. Instead, they should be interpreted as abstract logic statements. However, in the circuit diagrams, the signal names may be prefixed with the letter P or N to indicate whether the function is true at the upper or lower level, respectively. The P and N prefixes are normally not used for net names which start with P and N, respectively. Net numbers starting with 'n' or 'm' are true at the lower level and take the P prefix if true at the upper level. In the logic equations, the symbols ·, +, and ⊕ represent the Boolean AND, OR, and EXCLUSIVE OR functions, respectively. The apostrophe (') represents negation.

B. 5B/6B-T Encoding

In an exemplary embodiment, the 33 primary 6B vectors are classified into five groups as illustrated by the trellis diagrams of FIGS. 1A, 1B, 2, 3, and 4. All the coded vectors of FIGS. 2 through 4 have alternate, complementary versions (not shown in these figures but shown in FIG. 5) assigned to an identical source vector.

Conceptually, coding is generally performed in two steps. First, the translation to a primary vector is made. For the coded vectors of FIGS. 1A, 2, 3, and 4, the first five bits of the coded vector are identical to the source vector and the sixth and last bit assumes a default value of zero. Any vectors with changes in individual bits of the source vector belong to the balanced, disparity independent class of Fig. 1B. A second step for the subset of the disparity dependent coded vectors of FIGS. 2, 3, and 4 determines whether the alternate, complemented vector must be used to meet the disparity rules. Disparity dependent vectors have a plus sign or a minus sign in the DR column of the tables.

Generation of Primary 6B-T Vectors

The logic equations necessary for the translation to the primary vectors can be read directly from the columns 'Primary abcdei' and 'Primary fghj' of the tables shown in FIGS. 5 and 6, respectively. FIG. 5 shows a table for 5B/6B-T encoding, while FIG. 6 shows a table for 3B/4B-T encoding. In the 'Primary' columns of these tables, all plain bits are the same as the corresponding input bit values ABCDE or FGH, respectively. The bold and underlined bits are forced to the complemented value indicated. The i-bit and the j-bit have a default value of zero. For the new code, only the nine 6B-T vectors of FIG. 1B require any changes in individual bit

values as indicated by underscored bold entries and explicitly stated in the column 'Inverted Bits'.

The encoding equations are extracted from the encoding tables in methodical steps as described below. For each column of a coded bit such as 'a', the vectors which require changes are listed, the bits to be complemented have a superscript asterisk, and the bits which can be used to classify the source vector sets are generally presented in boldface type.

The 'a' column has bold entries for D0, D15, D16, and D31. The respective uncoded bits ABCDE are listed, the A-bit has a superscript asterisk, and common patterns are marked to logically classify the vectors by simple expressions.

D0	$0^{*}\mathbf{0\,0\,0}\,0$	D15	$1^{*}\mathbf{1\,1\,1}\,0$
D16	$0^{*}\mathbf{0\,0\,0}\,1$	D31	$1^{*}\mathbf{1\,1\,1}\,1$

Ignoring the 'A' bit, D0 and D16 can be identified as a class by $B' \cdot C' \cdot D'$.

D15 and D31 are identified by BCD.

Using these identifiers, the encoding equation for bit 'a' can be written as follows:

$$a = A \cdot (B \cdot C \cdot D)' + B' \cdot C' \cdot D'$$

The 'b' column has bold entries for D4, D8, D15, and D31.

D4	$\mathbf{0}\,0^{*}\,1\,0\,\mathbf{0}$	D15	$\mathbf{1}\,1^{*}\,\mathbf{1\,1}\,0$
D8	$\mathbf{0}\,0^{*}\,0\,1\,\mathbf{0}$	D31	$\mathbf{1}\,1^{*}\,\mathbf{1\,1}\,1$

Ignoring the 'B' bit, D4 and D8 can be identified as a class by $A' \cdot E' \cdot (C \neq D)$.

D15 and D31 are identified by ACD.

Using these identifiers, the encoding equation for bit 'b' can be written as follows:

$$b = B \cdot (A \cdot C \cdot D)' + A' \cdot E' \cdot (C \neq D).$$

The 'c' column has a bold entry for D1.

D1 $\quad 1\,0\,0^{*}\,0\,0$

The encoding equation for bit 'c' can be written as follows:

$$c = C + A \cdot B' \cdot D' \cdot E'$$

The 'd' column has bold entries for D0 and D31.

D0 $\quad 0\,0\,0\,0^{*}\,0$

D31 $\quad 1\,1\,1\,1^{*}\,1$

The encoding equation for bit 'd' can be written as follows:

$$d = D \cdot (A \cdot B \cdot C \cdot E)' + (A' \cdot B' \cdot C' \cdot E')$$

The 'e' column has a bold entry for D2.

D2 0 1 0 0 0*

The encoding equation for bit 'e' can be written as follows:

$$e = E + A' \cdot B \cdot C' \cdot D'$$

The 'i' column has nine entries with a value of one. All are marked in bold because the default value for the i-bit is zero.

D0	0 0 **0 0 0**	D0	**0 0 0 0** 0	D4	**0 0** 1 0 **0**
D1	1 0 **0 0 0**	D16	**0 0 0** 1	D8	**0 0** 0 1 **0**
D2	0 1 **0 0 0**			D15	**1 1 1** 1 0
D3	1 1 **0 0 0** K=0			D31	**1 1 1 1** 1

The first four entries D0 to D3 can be identified by $C' \cdot D' \cdot E' \cdot K'$.

D0 and D16 are identified by $A' \cdot B' \cdot C' \cdot D'$ (taking advantage of redundancy, since D0 is also listed in the left column).

D4 and D8 are identified by $A' \cdot B' \cdot E' \cdot (C \neq D)$.

D15 and D31 are identified by $A \cdot B \cdot C \cdot D$.

The encoding equation for bit 'i' can be written as follows:

$$i = C' \cdot D' \cdot E' \cdot K' + A' \cdot B' \cdot C' \cdot D' + A' \cdot B' \cdot E' \cdot (C \neq D) + A \cdot B \cdot C \cdot D$$

An important purpose for the S-function (e.g., S1 and S2 annotation in FIG. 5) is to prevent false commas in the bits 'cdeifgh'. In the original 8B/10B code of Franaszek and Widmer, U.S. Patent No. 4,486,739, incorporated by reference above, the vectors D11 and D31 (D20 in the original code) were included in the S-function only to lower the incidence of runs of five. They are left out now in order to reduce logic delay in the path for the encoding of the f and j bits, which have now delays comparable with the other bits. This simplifies and increases the effectiveness of pipe-lining, which is preferred for some high speed applications over parallel operation of several encoders for a single serial lane. The incidence of runs of five is now an average of about once per 256 bytes of random data, but there are still no contiguous runs of five.

For a positive running disparity at the front of the 6B vector (PDFS6), the S function should be asserted for the two balanced vectors for which the coded pattern ends with cdei=1100.

	Source Vector	Coded Vector
D13	1 0 **1 1 0**	1 0 1 1 0 0
D14	0 1 **1 1 0**	0 1 1 1 0 0

$$S1 = PDFS6 \cdot C \cdot D \cdot E' \cdot (A \neq B)$$

For a negative running disparity at the front of the 6B vector (NDFS6), the S function should be asserted for the two balanced vectors for which the coded pattern ends with cdei=0011.

	Source Vector	Coded Vector
D2	**0** 1 **0 0** 0	0 1 0 0 1 1
D16	**0 0 0 0** 1	1 0 0 0 1 1

$$S2 = NDFS6 \cdot A' \cdot C' \cdot D' \cdot (B \neq E)$$

C. 3B/4B-T Encoding

New 3B/4B-T vector assignments have been made following similar guidelines as used for the 5B/6B-T part. Changes are also made to accommodate the control character swap referred to above. FIG. 6 shows a coding table for the 3B/4B-T transmission code. The 3B/4B-T encoding equations can be directly derived from FIG. 6.

For the 'f' column, the coding equation is as follows:

$$f = F \cdot [F \cdot G \cdot H \cdot (S + Ky)]' = F \cdot [G \cdot H \cdot (S + Ky)]',$$

where $S = S1 + S2$ and $Ky = K23 + K27 + K29 + K30 = K \cdot K3' = K \cdot E$.

	Source Vector
K23	1 1 1 0 1 1
K27	1 1 0 1 1 1
K29	1 0 1 1 1 1
K30	0 1 1 1 1 1
K3	1 1 0 0 **0** 1

Ky includes also the control characters of FIG. 8, if implemented, and the simple equation Ky = KE still holds for six of those characters (K19.7, K21.7, K22.7, K25.7, K26.7, K28.7).

Because the primary interest here is minimum circuit delay rather than circuit area, the equation for bit 'f' is transformed as follows to reduce the logic depth:

$$f' = F' + G \cdot H \cdot S1 + G \cdot H \cdot S2 + G \cdot H \cdot K \cdot E$$

$$f' = F' + G \cdot H \cdot C \cdot D \cdot E' \cdot (A \neq B) \cdot PDFS6$$
$$+G \cdot H \cdot A' \cdot C' \cdot D' \cdot (B \neq E) \cdot NDFS6 + G \cdot H \cdot K \cdot E$$

For the actual circuit implementation, the term $G \cdot H$ is expanded back to $F \cdot G \cdot H$ because the full term is required for the j-bit encoding anyway.

For the 'g' column, the coding equation is as follows:

$$g = G + F' \cdot G' \cdot H' = G + F' \cdot H'$$

For the 'h' column, the coding equation is as follows:

h = H

For the 'j' column, the coding equation is as follows:

$$j = G' \cdot H' + F \cdot G \cdot H \cdot (S + Ky)$$

$$j = G' \cdot H' + F \cdot G \cdot H \cdot S1 + F \cdot G \cdot H \cdot S2 + F \cdot G \cdot H \cdot Ky$$

$$j = G' \cdot H' + F \cdot G \cdot H \cdot C \cdot D \cdot E' \cdot (A \neq B) \cdot PDFS6$$
$$+F \cdot G \cdot H \cdot A' \cdot C' \cdot D' \cdot (B \neq E) \cdot NDFS6 + F \cdot G \cdot H \cdot K \cdot E$$

D. Control Characters

A basic set of 12 control characters is listed in the table shown in FIG. 7. The coded format of all characters except K3.7 is identical to the traditional partitioned 8B/10B code. However, the source vector K28 is replaced by its complement K3 and so the name for the respective identical coded vectors is changed. K3.1 and K3.5 are comma characters and the comma sequence is printed in bold type. Note that because of the run length limit of five, the second bit (b) of the sequence can be left out for purposes of comma search circuits which limits the search to 1x00000 or 0x11111.

In the table of FIG. 7, the column heading 'Primary (6B)' refers to the fact that the 6B part of the 10-bit control vectors of that column are primary vectors. The 4B part may be the alternate vector. This distinction has no significance beyond semantics.

An additional set of nine control characters are defined in the table shown in FIG. 8. All these characters use the alternate A7 coding in the 4B domain when following a 6B vector with K = 1, which does not require the alternate code for purposes of compliance with the coding constraints. The 6B part is a disparity dependent balanced vector, which is complemented by

analogous rules as are the balanced 4B vectors when the K-bit has a value of one. These nine extra characters are not implemented in the circuit diagrams attached and treated as invalid characters. It is useful to know of their existence for certain applications.

E. Implementation of 8B/10B-T Bit Encoding

An implementation according to the above table shown in FIGS. 5 and 6, equations and design principles is illustrated in the circuit diagram shown in FIG. 9A. FIG. 9A shows an encoding portion of 8B/10B-T encoding circuitry. The design presented here assumes that all inputs are available in complementary form. PA, PB, PC, PD, and PE are from a five-bit source vector, from which the six-bit coded vector PCa, PCb, PCd, PCe, and PCi is created. PF, PH, and PG are from a three-bit source vector, from which four-bit coded vector PCf, PCg, PCh, and PCj is created.

In FIGS. 9A and 9B, there are a number of AND-OR (AO) blocks I152 and I429. There are also a number of AND-OR-INVERT (AOI) blocks I353, I348, I280, I354, I430, I474, and I469. An AO block contains a number of two-input AND gates, the outputs of which are coupled to an OR gate. An AOI block is similar, but the output of the OR gate is inverted. AO and AOI blocks are described in more detail in Widmer, PCT Application No. US02/13,798, entitled, "8B/10B Encoding and Decoding for High Speed Applications," filed on April 30, 2002 (claiming the benefit of U.S. Provisional Patent Application No. 60/289,556, filed on May 8, 2001), already incorporated by reference above.

The signal PCMPLS6 comes from the disparity control circuit shown in FIG. 9B. This signal is used to invert the primary coded vector abcdei to create the alternate coded vector (abcdei)'. This inversion is performed according to disparity rules. Similarly, the signal PCMPLS4 comes from the disparity control circuit shown in FIG. 9B below, and this signal is used to invert the primary coded vector fghj to create the alternate coded vector (fghj)' according to disparity rules.

Notation for net names in the encoding circuit diagrams: The letters 'a' and 'o' within net-names refer to the Boolean AND and OR functions, respectively. The letter 'n' within a name negates the preceding parameter. The letters 'e' and 'ue' represent the symbols '=' and '≠', respectively. The capital letters "ABCDEFGHK" represent the uncoded input bits and the

lower case letters "abcdeifghj" represent the coded format. These notations have been adopted because of the limitations of the logic design system.

F. 8B/10B-T Disparity Control

FIG. 9B shows a disparity control portion of the 8B/10B-T encoding circuitry. The circuit in FIG. 9B is used for computing disparity, in accordance with disparity rules. The column 'DR Class' (DR = required running disparity) in FIGS. 5 and 6 classifies the vectors according to the plus sign and the minus sign entries, which indicate the required disparity at the front of the primary coded vector. The expressions PDRS6, PDRS4 and NDRS6, NDRS4 represent a positive or negative required disparity, respectively, at the start of the 6B or 4B vectors. These signal names do not appear in the circuit diagram, because the gating required for CMPLS6 and CMPLS4 described below has been merged with said functions in order to eliminate one gating level.

1. PDRS6

The set of ten primary 6B vectors with a negative block disparity and a plus sign in the DR column of FIG. 5 is referred to as PDRS6. All are illustrated in the trellis diagram of FIG. 4. They are generated by appending a zero bit to the following source vectors:

D5	1 0 1 0 **0**	D12	**0** 0 1 1 0	D17	1 **0** **0** 0 **1**	
D6	0 1 1 0 **0**	D18	**0** 1 0 0 1	D24	0 **0** **0** 1 **1**	
D9	1 0 0 1 **0**	D20	**0** 0 1 0 1	K3	1 1 0 0 0	K=1
D10	0 1 0 1 **0**	D10	**0** 1 0 1 0			

The vectors D5, D6, D9, and D10 can be identified by the expression $E' \cdot (A \neq B) \cdot (C \neq D)$.

The vectors D12, D18, D20, and D10 (redundant) are identified by $A' \cdot (B \neq C) \cdot (D \neq E)$.

The vectors D17 and D24 are identified by $B' \cdot C' \cdot E \cdot (A \neq D)$.

The control vector K3 can be identified by $E' \cdot K$ since an examination of FIG. 5 shows that the other control characters K23, K27, K29, and K30 have all a value of one in bit position E.

The equation for PDRS6 can thus be expressed as follows:

$$PDRS6 = E' \cdot (A \neq B) \cdot (C \neq D) + A' \cdot (B \neq C) \cdot (D \neq E)$$
$$+B' \cdot C' \cdot E \cdot (A \neq D) + E' \cdot K$$

2. NDRS6

The set of five 6B vectors with a minus sign in the DR column of FIG. 5 is referred to as NDRS6. Four of these vectors have a positive block disparity and are illustrated in the trellis diagram of FIG. 3. The balanced vector D7 (111000) of FIG. 2 also requires a negative entry disparity.

D7	**1 1 1 0** 0	D/K29	1 0 **1 1 1**	D/K27	1 1 0 1 1
D/K23	**1 1 1 0** 1	D/K30	0 1 **1 1 1**		

The vectors D7 and D/K23 can be identified by $A \cdot B \cdot C \cdot D'$.

The vectors D/K29and D/K30 are identified by $C \cdot D \cdot E \cdot (A \neq B)$.

The vector D/K27 is identified by $A \cdot B \cdot C' \cdot D \cdot E$.

So the equation for NDRS6 is as follows:

$$NDRS6 = A \cdot B \cdot C \cdot D' + C \cdot D \cdot E \cdot (A \neq B) + A \cdot B \cdot C' \cdot D \cdot E$$

3. PDRS4

The table of FIG. 6 shows a plus sign in the DR column for the following six vectors shown with their uncoded values FGH K:

K3.0	0 0 0 **1**	Dx/K3.2	**0** 1 0	x	
K3.1	1 0 0 **1**	Dx/K3.4	**0** 0 1	x	
K3.5	1 0 1 **1**				
K3.6	0 1 1 **1**				

The four vectors in the left column above can be identified by $K \cdot (F \cdot G)'$ since none of the control characters have a DR entry of ± and all control characters with a negative DR have bit values of one for both the F and the G bit as shown in FIG. 6.

The two vectors of the right column are identified by $F' \cdot (G \neq H)$.

The equation for PDRS4 is as follows:

$$PDRS4 = K \cdot (F \cdot G)' + F' \cdot (G \neq H)$$

4. NDRS4

There are four rows in FIG. 6 with a minus entry in the DR column. They all can be uniquely identified by FG. Therefore:

$$NDRS4 = F \cdot G$$

5. CMPLS6 and CMPLS4

If the running disparity DF in front of the vector does not match the required entry disparity DR, a complement signal is generated which selects the alternate vector.

$$CMPLS6 = NDFS6 \cdot PDRS6 + PDFS6 \cdot NDRS6$$

$$CMPLS4 = NDFS4 \cdot PDRS4 + PDFS4 \cdot NDRS4$$

In the circuit diagrams, the signal names PDFS6, PDFS4 and NDFS6 and NDFS4 represent the actual running disparity at the front of the 6B and 4B vectors, respectively. Note that in the above two equations, the signals NDFS and PDFS are complementary and the signals PDR and NDR are orthogonal, i.e. only one can be true, but both can be false.

6. BALS6

The set of 19 primary 6B vectors of FIGS. 1A, 1B and 2 are balanced and identified by a 0 in the column 'DB Class' (i.e., block disparity). This set of vectors is referred to as BALS6 and can be grouped as shown below:

D0 00000	D8 00010	D25 10011	D0 00000
D1 10000	D11 11010	D26 01011	D4 00100
D2 01000	D13 10110	D28 00111	
D3 11000 K=0	D14 01110	D31 11111	D7 11100
	D16 00001		D15 11110
	D19 11001		
	D21 10101		
	D22 01101		

The 4 vectors D0, D1, D2, and D3 are identified by $C' \cdot D' \cdot E' \cdot K'$.

The 8 vectors D8, D11, D13, D14, D16, D19, D21, and D22 are identified by

$$XB6 = (D \neq E) \cdot (A' \cdot B' \cdot C' + A \cdot B \cdot C' + A \cdot B' \cdot C + A' \cdot B \cdot C)$$

$$= (D \neq E) \cdot (A \oplus B \oplus C)'.$$

The 4 vectors D25, D26, D28, and D31 are identified by

$$YB6 = D \cdot E \cdot (A \cdot B \cdot C + A \cdot B' \cdot C' + A' \cdot B \cdot C' + A' \cdot B' \cdot C)$$

$$= D \cdot E \cdot (A \oplus B \oplus C).$$

The 2 vectors D0 (redundant from Column 1) and D4 are identified by

$$WB6 = A' \cdot B' \cdot D' \cdot E'.$$

The 2 vectors D7 and D15 are identified by $ZB6 = A \cdot B \cdot C \cdot E'$.

So the equation for BALS6 can be expressed as:

$$BALS6 = (D \neq E) \cdot (A \oplus B \oplus C)' + D \cdot E \cdot (A \oplus B \oplus C)$$

$$+ A' \cdot B' \cdot D' \cdot E' + C' \cdot D' \cdot E' \cdot K' + A \cdot B \cdot C \cdot E'$$

In the usual circuit implementations, the signal BALS6 is in the critical delay path and required in true and complement form which requires an inversion with extra delay. This problem can be side-stepped by generating the UNBALS6 signal directly from the inputs similar to the circuit for BALS6. The UNBALS6 signal requires only nine gates and is slightly less complex and has a little less delay, so if only one of the signals is generated and then inverted, preference should be given to the UNBALAS6 signal. It is derived from the following grouping of all the vectors which have an entry other than 0 in the column DB of FIG. 5.

The 8 vectors D5, D9, D17, D29, D6, D10, D18, and D30 in the left column of the list below are identified by

$$XUB6 = (A \neq B) \cdot (C \cdot D' \cdot E' + C' \cdot D \cdot E' + C' \cdot D' \cdot E + C \cdot D \cdot E)$$

$$= (A \neq B) \cdot (C \oplus D \oplus E).$$

The three vectors D12, D20, and D24 in the second column can be identified by

$$YUB6 = A' \cdot B' \cdot (A \cdot B \cdot C' + A \cdot B' \cdot C + A' \cdot B \cdot C).$$

The two vectors D23 and D27 in the third column can be identified by

$$ZUB6 = A \cdot B \cdot E \cdot (C \neq D).$$

The single vector K3 in the right column can be uniquely identified by K=1 since all coded 6B K vectors are unbalanced.

D5	**1 0** 1 0 0	D12	**0 0** 1 1 0	D/K23	**1 1** 1 0 **1**	K3	1 1 0 0 0 0 K=1
D9	**1 0** 0 1 0	D20	**0 0** 1 0 1	D/K27	**1 1** 0 1 **1**		
D17	**1 0** 0 0 1	D24	**0 0** 0 1 1				
D/K29	**1 0** 1 1 1						
D6	**0 1** 1 0 0						
D10	**0 1** 0 1 0						
D18	**0 1** 0 0 1						
D/K30	**0 1** 1 1 1						

So the equation for UNBALS6 can be expressed as:

$$UNBALS6 = (A \neq B) \cdot (C \oplus D \oplus E) + A' \cdot B' \cdot (A \cdot B \cdot C' + A \cdot B' \cdot C + A' \cdot B \cdot C) + A \cdot B \cdot E \cdot (C \neq D) + K$$

*7. **BALS4***

The set of ten source vectors with a zero in the DB column of FIG. 6 is referred to as BALS4.

Dx/K3.0	**0 0 0**	Dx/K3.3	**1 1** 0	Dx/K3.6	0 **1 1**
Dx/K3.1	1 **0 0**	Dx/K3.5	1 0 1	Dx/K3.5	1 0 **1**

In the third column, the Dx/K3.5 vector pair is listed redundantly to simplify the logic expression below.

$$BALS4 = G' \cdot H' + F \cdot (G \neq H) + H \cdot (F \neq G)$$

Additional circuits governed by the signals BALS6 and BALS4 indicate the balance of a byte by the signal PBALBY which assumes the upper level for a balanced byte.

The technique for reducing the delay for the disparity function extending over one or more bytes as taught in Widmer, PCT No. US02/13798, entitled "8B/10B Encoding and Decoding for High Speed Applications," claiming the benefit of U.S. Patent Application No. 60/289,556 that was filed on May 8, 2001, the disclosure of which is already incorporated by reference above, apply equally to the code presented here and is summarized briefly again. See FIGS. 3B, 5, 6, 7, and 8 in Widmer, PCT No. US02/13798. A reduction in the combined delay of a 5B/6B (or 5B/6B-T) and a 3B/4B (or 3B/4B-T) encoder or of several 8B/10B (or 8B/10B-T)

encoders operating in parallel results from the methodology used to determine the disparity at any vector boundary as shown at the bottom of circuit diagram FIG. 9B and on the diagram FIG. 10 (described below). Given a starting disparity such as NDFBY (Negative Disparity in Front of a Byte), the running disparity at any subsequent vector boundary remains unchanged if the combined number of balanced S6 and S4 vectors between the two points is even, otherwise it assumes the complementary polarity. This is in contrast to the more obvious techniques which observe the disparity as it propagates from vector to vector. The expression NDFS4 represents a negative running disparity in front of the 4B vector.

The 8B/10B-T encoding circuitry of FIG. 9B has an output which indicates whether the coded 10-bit byte is balanced or not, but there is no output to indicate the ending disparity. Generally, the starting disparity for a vector is determined from the disparity of a prior reference point and the odd or even number of balanced vectors in between.

G. Staggered 5B/6B-T and 3B/4B-T Encoding and Decoding

Staggered timing of the 5B/6B-T and 3B/4B-T encoding and decoding circuits can be used to reduce the latency of transceivers which can be an issue for short connections. It also helps to reduce the performance requirements of the circuits.

Before consideration is given to pipe-lining either the 5B/6B-T or the 3B/4B-T encoder, the possibility of delaying the encoding of the 4B vectors with reference of the 6B vectors should be evaluated. The encoder output is typically fed to a serializer. If the serializer is a multiplexer of the commutator type, it is necessary to double latch some of the trailing bits so the updating of the 10-bit register does not interfere with the serialization, i. e. the first 6 bits 'abcdei' are updated while the last four bits 'fghj' are serialized and vice-versa. It is recommended that this double latching function be moved in front of the encoder with the bits 'FGHK' so the execution of 6B encoding and 4B encoding is staggered. Such an arrangement greatly reduces the timing demands for the generation of the S-function, the BALS6, and the UNBALS6 signals.

Analogous staggering of the 6B/5B-T and the 4B/3B-T decoding circuits can be a useful technique.

H. Disparity Circuit for 1-Byte Encoder, Fast Version

Several disparity circuits are presented now. The first disparity circuit that will be described is a faster version, which is faster relative to a slower version (described below). The fast version allows encoding of a byte to take place in a single cycle. This is described in more detail in Widmer, PCT Application No. US02/13,798, entitled, "8B/10B Encoding and Decoding for High Speed Applications," filed on April 30, 2002 (claiming the benefit of U.S. Provisional Patent Application No. 60/289,556, filed on May 8, 2001), which has already been incorporated by reference above. A circuit is shown in FIG. 10 for computing disparity. The circuit in FIG. 10 uses the encoding circuitry of FIGS. 9A and 9B (encompassed in module I3), along with XNOR2 (e.g., a two-input XNOR) gate I25. This circuit takes advantage of the fact that it is not necessary that the starting disparity must be known immediately for the encoding process. Since the evaluation of the running disparity at the end of a byte may be in the critical delay path, the final operations for determining the starting disparity PDFBY of the next byte are deferred to the next byte interval to be executed while initial bit encoding independent of the running disparity is performed. The cost of doing this is to pass along two parameters rather than just one to the next byte interval but it increases the timing margin by an amount equal to the delay of the XNOR2 gate I25.

The circuit diagram in FIG. 10 provides the starting disparity PDFBY and a coded byte disparity 'PBALBY' for the current byte based on these identical two parameters carried over from the preceding byte. At the end of each byte cycle, the signals PDFBY and PBALBY are stored in two latches with outputs PDFBY_LAST and PBALBY_LAST, respectively. The respective latches are not shown since their timing is identical to or closely related to the timing for the data output latches. These parameters are used for the computation of the starting disparity of the next byte. The signal PDFBY is at the upper level for a positive running disparity in front of the new byte.

There are two coded vectors per byte (6B, 4B). So if there is an odd number of balanced or unbalanced vectors between the start of the current byte and a previous byte boundary, the starting disparity for the current byte is the complement of the disparity at the reference point, otherwise it is the same.

I. Disparity Circuit for 1-Byte Encoder, Slower Version

To better illustrate the faster approach (e.g., shown in section H) to disparity operations, a slower disparity circuit is shown in FIG. 11 and is applicable where the higher performance is not needed. The ending disparity PDEBY is derived within one and the same encoding cycle. Then only one parameter must be passed on to the next cycle with a single latch. The data input of this latch is PDEBY and the output is PDFBY, the disparity at the front of the next byte.

J. Disparity Circuit for 4-Byte Encoder, Faster Version

A disparity circuit is shown in FIG. 12 for determining disparity for four bytes. This circuit shows four encoders operating in parallel on a 4-byte word. The modules I1, I44, I38, and I3 each encompass the encoding circuitry of FIGS. 9A and 9B. The starting disparity of the block and of the first byte is given by the input PDFW (Pos. Disp. in Front of the Word). It is generated from signals carried over from the preceding word cycle by a pair of latches (not shown), i.e., the running disparity PDF3_LAST in front of the last byte (#3) and the balance signal PBALBY3_LAST of byte #3. The starting disparity for each of the remaining three bytes is obtained by circuits operating in parallel from this reference point and the number of balanced bytes in between using a set of XOR and XNOR gates. The signal PBAL012 is at the upper level if the block comprising the first three bytes is balanced.

K. Disparity Circuit for 4-Byte Encoder, Slower Version

A disparity circuit is shown in FIG. 13 for determining disparity for four bytes. The circuit shown in FIG. 13 is slower than the circuit shown in FIG. 12. In FIG. 13, the ending disparity PDEW of the word is generated within a single clock cycle. A single latch (not shown) with the signal PDEW at the data input passes along to the next cycle the ending disparity for the word. The output of this latch is the starting disparity PDFW for the next cycle.

L. 10B/8B-T Decoder

A 10B/8B-T decoder comprises circuits to restore the original byte ABCDEFGH K, and circuits to indicate all transmission errors to the extent that they are detectable by the transmission code. For decoding, the trailing bits 'i' and 'j' are just dropped but their value guides some of the decoding functions.

The tables in FIGS. 14 and 16 (described below) show the relationships between the coded and decoded vectors for the 6B/5B-T and 4B/3B-T decoding, respectively. For some vector names, there are several rows to represent the true and complement version and to show the different rules for the decoding of specific bits. If for a row, the equation in the column 'Decoding Class' is true, the bold, underlined bits in the column 'ABCDE K' are complemented. The suffix 'P' or 'A' of a vector name refers to the primary or alternate version of a vector, respectively.

1. 6B/5B-T Decoder

The relationship between the coded 6B vectors and the corresponding decoded 5B vectors for 6B/5B-T decoding, performed by a 6B/5B-T decoder is shown in FIG. 14. A 6B/5B-T bit decoding circuit is shown in FIG. 15A. FIG. 15B shows the corresponding disparity and error checking circuits of the decoder.

a. Inversion of the Five Leading Bits, CMPL5

If the received trailing bit 'i' has a value of one and the vector is not balanced or if the leading three bits have all a value of zero, then all the leading five bits are complemented. The complements of all valid vectors falling into this category are illustrated in FIGS. 2, 3, and 4 and their true values are listed in the 'Alternate' column of FIG. 5.

It is assumed that invalid vectors with five or six ones originated from vectors with four ones and they will be complemented as though the extra ones were not present, i.e., it is not necessary to include the zeros in the Boolean expressions for the 11 vectors below with positive disparity.

D5	0 1 0 1 1 1	D12	1 1 0 0 1 1	D9	0 1 1 0 1 1
D6	1 0 0 1 1 1	D20	1 1 0 1 0 1	D10	1 0 1 0 1 1
D7	0 0 0 1 1 1	D24	1 1 1 0 0 1	D17	0 1 1 1 0 1
K3	0 0 1 1 1 1	D18	1 0 1 1 0 1		

Similarly, it is assumed that invalid vectors with five or six zeros originated from vectors with four zeros and they will be complemented as though the extra zeros were not present, i.e., it is not necessary to include the ones in the Boolean expressions for the four vectors below with negative disparity.

D/K23 **0 0** 0 1 **0** 1	D/K29 0 1 **0 0 0** 1
D/K27 **0 0** 1 0 **0** 1	D/K30 1 0 **0 0 0** 1

A Boolean expression CMPL5 for the complementation of the leading five bits of the fifteen 6B vectors above is developed below:

$$CMPL5 = d \cdot e \cdot i + a \cdot b \cdot i \cdot (c + d + e) + c \cdot i \cdot (a + b) \cdot (d + e) + a' \cdot b' \cdot e' \cdot (c' + d') + c' \cdot d' \cdot e' \cdot (a' + b')$$

In the circuit diagram shown in FIG. 15A, the following net names are used:

$$n01 = a \cdot b \cdot i \cdot (c + d + e) + d \cdot e \cdot i \qquad n02 = c \cdot i \cdot (a + b) \cdot (d + e)$$

$$n1 = a' \cdot b' \cdot e' \cdot (c' + d') + c' \cdot d' \cdot e' \cdot (a' + b')$$

b. Selected Bit Inversions

There are nine disparity independent vectors with an i-bit value of one. For eight of these, one or more bits in the leading five positions must be changed for decoding.

Bit A

FIG. 14 shows for which of these vectors, the first bit should be complemented:

D0 1 **0** 0 1 0 **1**

D15 0 **0** 1 1 0 **1**

D16 1 **0** 0 0 1 **1**

D31 0 **0** 1 0 1 **1**

$$CMPLa = CMPL5 + b' \cdot i \cdot (a \neq c) \cdot (d \neq e)$$

$$n2 = b' \cdot i \cdot (a \neq c) \cdot (d \neq e)$$

Bit B

D4 **0 1** 1 0 **0 1** D15 **0 0** 1 1 0 **1**

D8 **0 1** 0 1 **0 1** D31 **0 0** 1 0 1 **1**

$$CMPLb = CMPL5 + a' \cdot b \cdot e' \cdot i \cdot (c \neq d) + a' \cdot b' \cdot c \cdot i \cdot (d \neq e)$$

$$n4 = a' \cdot b' \cdot c \cdot i$$

$$n5 = a' \cdot b \cdot e' \cdot i \cdot (c \neq d) + a' \cdot b' \cdot c \cdot i \cdot (d \neq e)$$

Bit C

D1 1 0 1 0 0 1

$CMPLc = CMPL5 + a \cdot b' \cdot c \cdot d' \cdot e' \cdot i$

Bit D

D0 1 **0** 0 1 0 **1**

D31 0 **0** 1 0 1 **1**

$CMPLd = CMPL5 + b' \cdot i \cdot (a \neq c) \cdot (c \neq d) \cdot (d \neq e)$

$n7 = b' \cdot i \cdot (a \neq c) \cdot (c \neq d) \cdot (d \neq e)$

Bit E

D2 0 1 0 0 1 1

$CMPLe = CMPL5 + a' \cdot b \cdot c' \cdot d' \cdot e \cdot i$

Bit K

See the table in FIG. 7: K3.x and Ky.7, where x is any number from zero to seven and y has a value of 23, 27, 29, or 30.

$K = (c = d = e = i) + (e \neq i) \cdot (i = g = h = j)$

The above equation is implemented as follows, because the term $(c' \cdot d' \cdot e' \cdot i')$ is also required in the 4B/3B-T decoder:

$K = c \cdot d \cdot e \cdot i + c' \cdot d' \cdot e' \cdot i' + (e \neq i) \cdot (i = g) \cdot (g = h) \cdot (h = j)$

c. Logic Equation for invalid Vectors R6, INVR6

There are a total of 16 invalid R6 vectors:

$INVR6 = a \cdot b \cdot c \cdot d + a' \cdot b' \cdot c' \cdot d' + P3x \cdot e \cdot i + Px3 \cdot e' \cdot i'$

$P3x = P31 + P40 = a \cdot b \cdot c + a \cdot b \cdot d + a \cdot c \cdot d + b \cdot c \cdot d$

$P3x = P13 + P04 = a' \cdot b' \cdot c' + a' \cdot b' \cdot d' + a' \cdot c' \cdot d' + b' \cdot c' \cdot d'$

Concerning notation for net names in the decoding circuit diagrams of FIGS. 15A and 15B: For the Boolean operators, the identical letters are use as for the encoding diagrams, but they are capitalized (A, O, N, E, UE) to avoid confusion with some of the lower case letters abcdeifghj which represent the coded bits.

2. 6B/5B-T Disparity Checks

The column 'DR' of the table shown in FIG. 14 lists the required disparity at the start of the respective 6B vectors and the column 'DR Class' identifies the respective input bit

patterns. In an application by Widmer, PCT Application No. US02/13,798, entitled, "8B/10B Encoding and Decoding for High Speed Applications," filed on April 30, 2002 (claiming the benefit of U.S. Provisional Patent Application No. 60/289,556, filed on May 8, 2001), incorporated by reference previously, the column DU is called DB. The old column DB has been changed to DU and lists a positive or negative exit disparity for the disparity dependent vectors only. Disparity independent vectors have no entry in the DU column. In the old design, disparity independent vectors passed the input disparity to the output. In the new design, disparity independent vectors are ignored and bypassed for disparity purposes for shorter delay. Short delay is especially important for the DU outputs PDUR6 and NDUR6. To achieve the shorter delay, a dedicated column 'DU Class' has been added, which sorts the received vectors into DU classes in the most efficient way.

a. Logic Equations for Required Input Disparity DRR6

The terms PDRR6 and NDRR6 represent the R6 vectors which require a positive or negative running disparity, respectively, at the start of the vector. All invalid vectors with five or six zeros are also assigned a positive required entry disparity. Therefore, any vector with three leading zeros or four or more zeros requires a positive entry disparity. The valid vectors with positive required front end disparity are listed below.

D7	**000**111	D5	101**000**	D9	100100
D/K23	**000**101	D6	011**000**	D10	010100
D24	**000**110	K3	110**000**	D12	001100
				D17	100010
				D18	010010
				D20	001010
				D/K27	001001
				D/K29	010001
				D/K30	100001

Making allowance for invalid vectors, the three vectors in the left column can be identified by the Boolean expression $a' \cdot b' \cdot c'$. The three vectors in the center column are identified by $d' \cdot e' \cdot i' \cdot (a \cdot b \cdot c)'$. The nine vectors in the right column have two zeros in both the leading and trailing three bit positions. Because of the inclusion of the vectors with more than

four zeros, all the one bits can be ignored and the remaining vectors (valid or invalid) can be identified by the expression:

$$(a' \cdot b' + a' \cdot c' + b' \cdot c') \cdot (d' \cdot e' + d' \cdot i' + e' \cdot i')$$

Therefore:

$$PDRR6 = a' \cdot b' \cdot c' + d' \cdot e' \cdot i' \cdot (a \cdot b \cdot c)' + (a' \cdot b' + a' \cdot c' + b' \cdot c') \cdot (d' \cdot e' + d' \cdot i' + e' \cdot i')$$

In the circuit diagram shown in FIG. 15B, the following net names are used:

$$n9 = (a' \cdot b' + a' \cdot c' + b' \cdot c')$$

$$n10 = (d' \cdot e' + d' \cdot i' + e' \cdot i')$$

$$n11 = n9 \cdot n10$$

$$n12 = d' \cdot e' \cdot i' \cdot (a \cdot b \cdot c)'$$

The vectors with negative required front end disparity are listed below.

D7	1 1 1 0 0 0	D5	0 1 0 1 1 1	D9	0 1 1 0 1 1
D/K23	1 1 1 0 1 0	D6	1 0 0 1 1 1	D10	1 0 1 0 1 1
D24	1 1 1 0 0 1	K3	0 0 1 1 1 1	D12	1 1 0 0 1 1
				D17	0 1 1 1 0 1
				D18	1 0 1 1 0 1
				D20	1 1 0 1 0 1
				D/K27	1 1 0 1 1 0
				D/K29	1 0 1 1 1 0
				D/K30	0 1 1 1 1 0

The three vectors D7, D/K23, and D24 in the left column above can be identified by the Boolean expression $a \cdot b \cdot c$. The three vectors in the center column are identified by $d \cdot e \cdot i \cdot (a + b + c)$. The nine vectors in the right column are identified by the expression:

$$(a \cdot b + a \cdot c + b \cdot c) \cdot (d \cdot e + d \cdot i + e \cdot i)$$

Therefore:

$$NDRR6 = a \cdot b \cdot c + d \cdot e \cdot i \cdot (a + b + c) + (a \cdot b + a \cdot c + b \cdot c) \cdot (d \cdot e + d \cdot i + e \cdot i)$$

$$n13 = (a \cdot b + a \cdot c + b \cdot c)$$

$$n14 = (d \cdot e + d \cdot i + e \cdot i)$$

$$n15 = d \cdot e \cdot i \cdot (a + b + c)$$

$$n16 = n13 \cdot n14$$

b. Logic Equation for Monitoring Byte Disparity Violations, DVBY

Bytes with only disparity independent vectors R6 and R4 are ignored for disparity checking purposes. There is a disparity violation DVBY at a specific byte under the following conditions:

1. The required entry disparity of the R6 vector does not match the running disparity at the front of the byte.

2. The required entry disparity of the R4 vector does not match the running disparity in front of the byte and the R6 vector does not have a required entry disparity which is the complement of the required entry disparity for R4.

$$n17 = NDRR4 \cdot PDFBY \cdot PDRR6'$$

$$n18 = PDRR4 \cdot NDFBY \cdot NDRR6'$$

A disparity violation internal to a byte from a disparity dependent R4 vector mismatched to a disparity dependent R6 vector is included in the set of invalid bytes, but not in DVBY. The disparity violation at a byte DVBY is thus given by the equation:

$$DVBY = NDFBY \cdot (PDRR6 + PDRR4 \cdot NDRR6') + PDFBY \cdot (NDRR6 + NDRR4 \cdot PDRR6')$$

The terms PDFBY and NDFBY represent a positive or negative running disparity, respectively, at the front of the byte and one or the other function is always true. However, for PDRR6 and NDRR6, none of the functions is true for the case of most balanced vectors.

c. Logic Equations for the assumed ending Disparities PDUR6 and NDUR6

Four leading ones or zeros in the encoded domain are invalid vectors and can be generated only by at least one error. For the case of a single error and e=i, the R6 vector was obviously one of the initially balanced vectors 011100, 101100, 110100, 111000, or their complement. Of these, all except 111000 and 000111 should not generate PDUR6 or NDUR6 which would generate a superfluous code violation at the next disparity dependent vector in addition to the invalid vector at the actual error location.

Therefore, the signal NDUR6 should be asserted in response to the following 6B inputs:

1. All bits with a value of zero (1).

2. Five bits with a value of zero (6).

3. Four bits with a value of zero except the pattern '000011' (14).

4. The pattern '111000'.

The list of these vectors is almost identical to the list above for PDRR6 except that the pattern for D7 is the complement, i.e., 111000, so the modified left column looks as follows:

D7 **1 1 1** 0 0 0

D/K23 **0 0 0 1** 0 1

D24 **0 0 0 1** 1 0

The vectors D/K23 and D24 above are defined by the expression:

$$n19 = a' \cdot b' \cdot c' \cdot d \cdot (e' + i')$$

The vector D7 is defined by the expression:

$$n20 = a \cdot b \cdot c \cdot d' \cdot e' \cdot i'$$

The following net name abbreviation is used in the circuit diagram shown in FIG. 15B:

$$n21 = n19 + n20$$

The expression for n21 replaces the term $a' \cdot b' \cdot c'$ in the PDRR6 equation, therefore:

$$NDUR6 = n21 + d' \cdot e' \cdot i' \cdot (a \cdot b \cdot c)' + (a' \cdot b' + a' \cdot c' + b' \cdot c') \cdot (d' \cdot e' + d' \cdot i' + e' \cdot i')$$

The signal PDUR6 should be asserted in response to the following 6B inputs:

1. All bits with a value of one (1).

2. Five bits with a value of one (6).

3. Four bits with a value of one except the pattern '111100' (14).

4. The pattern '000111'.

Again, the list of these vectors is almost identical to the list above for NDRR6 except that the pattern for D7 is the complement, i.e. 000111, so the left column looks as follows:

D7 **0 0 0** 1 1 1

D23 **1 1 1** 0 1 0

K24 **1 1 1** 0 0 1

The three vectors above are defined by the expression n24. The following net name abbreviations are used in the circuit diagram show in FIG. 15B:

$$n22 = a \cdot b \cdot c \cdot d' \cdot (e + i)$$

$$n23 = a' \cdot b' \cdot c' \cdot d \cdot e \cdot i$$

$$n24 = n22 + n23$$

The term n24 replaces the term $a \cdot b \cdot c$ in the NDRR6 equation:

Therefore:

$$PDUR6 = n24 + d \cdot e \cdot i \cdot (a + b + c) + (a \cdot b + a \cdot c + b \cdot c) \cdot (d \cdot e + d \cdot i + e \cdot i)$$

d. Circuit Simplification

The first term (D7) in each of the equations for PDUR6 and NDUR6 prevents double counts for some type of errors. However, overall some double counts are unavoidable and the added term improves the accuracy of the error count by a minuscule amount. It is debatable whether it should be dropped. A further more significant simplification is possible if the first vector (D7) is also dropped from PDRR6 and NDRR6. This may delay the error detection for some patterns by a very few bytes but does not degrade error detection per se. The circuit advantage is that with these two simplifications PDRR6 is equal to NDUR6, and NDRR6 is equal to PDUR6. Analogous simplifications are possible for the 4B/3B-T error detection, but there the circuit simplification is less compelling.

3. 4B/3B-T Decoder and Error Checks

The table in FIG. 16 shows relationships between the coded 4B vectors and the corresponding decoded 3B vectors. A decoder for 4B/3B-T decoding is shown in FIG. 17.

a. Logic Equations for the Generation of the decoded Bits F, G, H, K

Generally, F = f, G = g, H = h, except for the conditions listed below for which the complement of the respective uncoded bit is generated, e.g., H = h'.

Bit F

For 4B/3B-T decoding, the f-bit is complemented for the vectors listed below. For the four vectors in the left column, complementation is applicable only if the vectors are preceded by K3 with negative ending disparity, i.e., if $c' \cdot d' \cdot e' \cdot i'$ is true.

K3.0	1 0 1 0	Dx/K3.2	1 0 1 1	Dx/K3.7	0 0 0 1
K3.1	0 1 1 0	Dx/K3.3	0 0 1 1	Dx/K3.4	1 1 0 1
K3.5	0 1 0 1	Dx/Ky.7	0 1 1 1		
K3.6	1 0 0 1				

The left column can be characterized by $m0 = (c' \cdot d' \cdot e' \cdot i') \cdot (f \neq g) \cdot (h \neq j)$.

The center column can be characterized by $m2 = h \cdot j \cdot (f \cdot g)'$.

The right column can be characterized by $m1 = (f = g) \cdot h' \cdot j$.

The Boolean expression CMPLf to complement the f-bit is thus:

$$CMPLf = (c' \cdot d' \cdot e' \cdot i') \cdot (f \neq g) \cdot (h \neq j) + h \cdot j \cdot (f \cdot g)' + (f = g) \cdot h' \cdot j = m0 + m1 + m2$$

Bit G

For 4B/3B-T decoding, the g-bit is complemented for the vectors listed below. For the three vectors in the left column, complementation is applicable only if the vectors are preceded by K3 with negative ending disparity, i.e. if $c' \cdot d' \cdot e' \cdot i'$ is true.

K3.1	0 1 1 0	Dx/K3.2	1 0 1 1	Dx/K3.7	0 0 0 1
K3.5	0 1 0 1	Dx/K3.3	0 0 1 1	Dx/Ky.7	1 0 0 0
K3.6	1 0 0 1	Dx/K3.0	0 1 0 1		
		Dx/K3.4	1 1 0 1		

The left column can be characterized by

$$(c' \cdot d' \cdot e' \cdot i') \cdot (f \neq g) \cdot (h \neq j) \cdot (f \cdot h)' = m0 \cdot (f \cdot h)'$$

The center column can be characterized by $m5 = g' \cdot h \cdot j + g \cdot h' \cdot j$.

The right column can be characterized by $m4 = (f \neq j) \cdot g' \cdot h'$.

The Boolean expression CMPLg to complement the f-bit is thus:

$$CMPLg = (c' \cdot d' \cdot e' \cdot i') \cdot (f \neq g) \cdot (h \neq j) \cdot (f \cdot h)' + g' \cdot h \cdot j + g \cdot h' \cdot j + (f \neq j) \cdot g' \cdot$$

$$CMPLg = m0 \cdot (f \cdot h)' + m4 + m5$$

Bit H

For 4B/3B-T decoding, the h-bit is complemented for the vectors listed below. For the four vectors in the left column, complementation is applicable only if the vectors are preceded by K3 with negative ending disparity, i.e., if $c' \cdot d' \cdot e' \cdot i'$ is true.

K3.0	1 0 1 0	Dx/K3.2	1 **0 1 1**	Dx/K3.7	0 0 0 **1**
K3.1	0 1 1 0	Dx/K3.3	0 **0 1 1**	Dx/K3.4	1 1 **0 1**
K3.5	0 1 0 1			Dx/Ky.7	1 0 0 0
K3.6	1 0 0 1				

The left column is identical to what is listed under Bit F above and can be characterized by $m0 = (c' \cdot d' \cdot e' \cdot i') \cdot (f \neq g) \cdot (h \neq j)$.

The center column can be characterized by $g' \cdot h \cdot j$.

The right column can be characterized by $(f = g) \cdot h' \cdot j + (f \neq j) \cdot g' \cdot h' = m1 + m4$, where the second term includes Dx/K3.7 redundantly to reuse an expression already available from bit g decoding.

The Boolean expression CMPLh to complement the h-bit is thus:

$$CMPLh = (c' \cdot d' \cdot e' \cdot i') \cdot (f \neq g) \cdot (h \neq j) + g' \cdot h \cdot j + (f = g) \cdot h' \cdot j + (f \neq j) \cdot g' \cdot h'$$

$$CMPLh = m0 + m1 + m4 + g' \cdot h \cdot j$$

b. Logic Equations for the required Disparity at the Front of the R4 Vector

The terms PDRR4 and NDRR4 represent the required positive or negative disparity, respectively, at the front of the R4 vector.

A total of six 4B vectors require a positive disparity PDRR4 at the front:

1. All four bits have a zero value which is an invalid vector.
2. There is a single one bit in the 4-bit vector.
3. The vector 0011.

The first and the third condition are met by $f' \cdot g'$ which also overlaps the second condition. The second condition is met by $m6 = (f \cdot g)' \cdot h' \cdot j'$. Therefore,

$$PDRR4 = f' \cdot g' + m6$$

A total of six 4B vectors require a negative disparity NDRR4 at the front:

1. All four bits have a one value which is an invalid vector.
2. There is a single zero bit in the 4-bit vector.
3. The vector 1100.

$$NDRR4 = f \cdot g + (f + g) \cdot h \cdot j$$

c. Logic Equations for the assumed ending Disparities PDUR4 and NDUR4

A total of six 4B vectors have a positive ending disparity PDUR4:

1. All four bits have a one value which is an invalid vector,

2. There is a single zero bit in the 4-bit vector.

3. The vector 0011.

$$PDUR4 = h \cdot j + f \cdot g \cdot (h + j)$$

A total of six 4B vectors have a negative ending disparity NDUR4:

1. All four bits have a zero value which is an invalid vector.

2. There is a single one bit in the 4-bit vector.

3. The vector 1100.

$$NDUR4 = h' \cdot j' + f' \cdot g' \cdot (h \cdot j)'$$

d. Logic Equation for invalid Vector R4, INVR4

There are a total of two inherently invalid R4 vectors: all ones or all zeros ($f = g = h = j$). Some combinations of R6 and R4 vectors are invalid, such as violations of the S-function rules ($c \neq e = i = f = g = h$), which represents a false comma, and the bit configuration ($i \neq g = h = j$) which can generate another false comma. These invalid R4 vectors are lumped together in the signal INVR4:

$$INVR4 = (f = g = h = j) + (c \neq e = i = f = g = h) + (i \neq g = h = j)$$

The nine Kx.7 control characters shown in the table in FIG. 8 are not implemented in the circuit designs shown and therefore classified as invalid characters. The R6 part of these characters is derived from 5-bit source vectors of the P22 class, which have two ones and two zeros in the first four bit positions, and additionally they have complementary bits in the last two positions. So an invalid Kx.7 control character is present, if the following conditions are met:

$$VKx7 = (i \cdot g \cdot h \cdot j + i' \cdot g' \cdot h' \cdot j') \cdot (e \neq i) \cdot P22$$

Where $P22 = (a \neq b) \cdot (c \neq d) + (a = b) \cdot (c = d) \cdot (b \neq c)$

If the required entry disparity DRR4 does not match the exit disparity DUR6 of a disparity dependent R6 vector, an invalid character is flagged by the signal DV64 under the following conditions:

$$DV64 = PDRR4 \cdot NDUR6 + NDRR4 \cdot PDUR6$$

The signal INVR4V64 is the OR function of INVR4 and DV64.

4. 10B/8B-T Decoder, Error Checks

A 10B/8B-T decoder is shown in FIG. 18.

a. Error Reporting

The circuit shown in FIG. 18 merges the 6B/5B-T and 4B/3B-T decoders into a byte decoder and generates the signal INVBY, which indicates an inherently invalid byte that includes disparity violations DV64. These violations are evident from an examination of just the 10 coded bits of the current byte. The output INVBY is provided for support of certain error correction techniques which require the identification of the erroneous byte. The signal VIOL signals either an invalid byte or a disparity violation DVBY detected at this location which may result from an error in this or a preceding byte. The circuit delays for DVBY and VIOL are the longest for the decoder. Fortunately, for many applications these outputs can be reported during the next byte cycle with no adverse impact, i.e., pipe-lining limited to these circuits is possible.

b. Disparity Monitoring

If either one or both of the vectors are disparity dependent, either PDUBY or PNDUBY are asserted to establish a positive or negative running disparity, respectively, at the end of the byte:

$$PDUBY = PDUR4 + PDUR6 \cdot NDUR4'$$

$$NDUBY = NDUR4 + NDUR6 \cdot PDUR4'$$

5. Byte Disparity, Fast Version

FIG. 19 shows circuitry for computing disparity for a single byte by using the decoder of FIG. 18. This circuit computes the disparity in a relatively fast manner, as compared to a circuit described below. Notation is as follows: The signal names PDFBY and NDFBY refer to a positive and negative running disparity, respectively, in front of the byte. The signal names PDUBY and NDUBY refer to the assumed positive or negative exit disparity, respectively, of the byte, regardless of the starting disparity at the front end. If neither the 6B vector nor the 4B vector of the byte is disparity dependent, none of the two outputs is asserted.

The values for the outputs NPDFBY, PDUBY, and PNDUBY are stored in three latches (not shown in the diagrams) which are clocked concurrently with the decoded data output. The outputs of the latches are labelled NPDFBY_LAST, NPDUBY_LAST (from pin

L2N, the inverted output of the slave latch L2), and PNDUBY_LAST, respectively, and are used for the computation of the starting disparity PDFBY for the next byte.

6. Logic Equations for the Determination of the Disparity at the Start of the Byte

$$PDFBY = PDUBY\_LAST + PDFBY\_LAST \cdot NDUBY\_LAST'$$

Note that NDFBY and PDFBY are complementary: NDFBY = PDFBY' and the values of PDUBY and NDUBY are exclusive, none or one alone can be true.

7. Byte Disparity, Slower Version

FIG. 20 shows circuitry for computing disparity for a single byte by using the decoder of FIG. 18. This circuit computes the disparity in a relatively slow manner. The incentive to use the slower version is the saving of two latches. If timing is not critical, the ending disparity PDEBY is generated in the same cycle as the decoding and the error checks. So only this single parameter must be passed on to next byte in the traditional manner. The output of this latch is the signal PDFBY, the disparity in front of the new byte. If the longest delay path is to the PDEBY output, the critical path delays have then been increased by one inverter plus one OAI21 gate delay.

8. Four-Byte Word Decoder

A four-byte word decoder is shown in FIG. 21.

a. Notation

The signal names PDFBY0 and NDFBY0 refer to a positive and negative disparity, respectively, in front of byte #0.

The signal names PDUBY0 and NDUBY0 refer the assumed positive or negative exit disparity, respectively, of byte #0. If neither the 6B vector nor the 4B vector of the byte is disparity dependent, none of the two outputs is asserted.

The values for the outputs PDFBY3, PDUBY3, and PNDUBY3 are stored in the latches with the complementary outputs PDFBY3_LAST/NPDFBY3_LAST, PDUBY3_LAST/ NPDUBY3_LAST, and PNDUBY3_LAST/NDUBY3_LAST, respectively, for the computation of the starting disparities PDFBY0 and PNDFBY0 for the next word cycle at the top of the diagram.

b. Logic Equations for the Determination of the Disparity at the Start of the Bytes

$$PDFBY0 = PDUBY3\_LAST + PDFBY3\_LAST \cdot NDUBY3\_LAST'$$

$$NDFBY0 = NDUBY3\_LAST + NDFBY3\_LAST \cdot PDUBY3\_LAST'$$

The values of PDFBY0 and NDFBY0 are complementary, but the values of PDUBY3 and NDUBY3 are exclusive, none or one alone can be true.

To minimize the circuit delays, the disparity values for the front of byte #1, #2, and #3 are determined not sequentially from byte to byte, but based on the disparity in front of byte #0 and the changes in disparity contributed by the byte(s) in between.

$$PDFBY1 = PDUBY0 + PDFB0 \cdot NDUB0'$$

$$NDFBY1 = NDUBY0 + NDFB0 \cdot PDUBY0'$$

$$PDFBY2 = PDUBY1 + PDUB0 \cdot NDUB1' + PDFBY0 \cdot NDUBY0' \cdot NDUBY1'$$

$$n0 = NDUBY0 + NDUBY1$$

$$PDFBY2 = PDUBY1 + PDUB0 \cdot NDUB1' + PDFBY0 \cdot n0'$$

$$NDFBY2 = NDUBY1 + NDUBY0 \cdot PDUBY1' + NDFBY0 \cdot PDUBY0' \cdot PDUBY1'$$

$$n1 = PDUBY0 + PDUBy1$$

$$NDFBY2 = NDUBY1 + NDUBY0 \cdot PDUBY1' + NDFBY0 \cdot n1'$$

$$PDFBY3 = PDUBY2 + PDUBY1 \cdot NDUBY2' + PDUBY0 \cdot NDUBY1' \cdot NDUBY2' + PDFBY0 \cdot NDUBY0' \cdot NDUBY1' \cdot NDUBY2'$$

$$n2 = NDUBY0 + NDUBY1 + NDUBY2$$

$$n3 = NDUBY1 + NDUBY2$$

$$PDFBY3 = PDUBY2 + PDUBY1 \cdot NDUBY2' + PDUBY0 \cdot n3' + PDFBY0 \cdot n2$$

$$NDFBY3 = NDUBY2 + NDUBY1 \cdot PDUBY2' + NDUBY0 \cdot PDUBY1' \cdot PDUBY2' + NDFBY0 \cdot PDUBY0' \cdot PDUBY1' \cdot PDUBY2'$$

$$n4 = PDUBY0 + PDUBY1 + PDUBY2$$

$$n5 = PDUBY1 + PDUBY2$$

$$NDFBY3 = NDUBY2 + NDUBY1 \cdot PDUBY2' + NDUBY0 \cdot n5 + NDFBY0 \cdot n4'$$

In the circuit implementation of FIG. 21, the following relationships may be taken advantage of:

$$PNDFBY1 = PDFBY1'$$

$$PNDFBY2 = PDFBY2'$$

These signals are not in the critical path and the added inversion does not decrease the maximum rate. For applications which have sufficient timing margin, the same simplifications can also be used for the signal PNDFBY3 and perhaps PNDFBY0 at a penalty of one inversion for each of those two signals.

M. 5B/6B-T Code and Partitioned 10B/12B Code

Some applications are compatible with modulo 5-bit data units. Plain data can be transmitted in modulo five format by just concatenating 6B vectors as defined above. However, in the 6-bit domain there is no comma character defined. Synchronization must be accomplished by other means. To gain access to comma characters, the code should be viewed as a 10B/12B code, but actual operation can easily shift from one mode to the other. A compatible partitioned 10B/12B transmission code can be constructed from concatenated pairs of 6B vectors with identical definitions as defined in FIG. 5. A difference between a 5B/6B-T and the 10B/12B code is the availability of a comma and a larger set of control characters in the 12B format. For purposes of special, non-data characters, for example a comma, the 5B/6B-T code is expanded to a 10B/12B code using a pair of contiguous 6B vectors (2 x 5B/6B-T = 10B/12B). For an example, refer to FIG. 22, where ($a_0b_0c_0d_0e_0i_0$) and ($a_1b_1c_1d_1e_1i_1$) are needed to define these characters. The comma-character is a special character of interest. It can be used to mark and recover quickly, and to monitor the 6B, 12B, and packet boundary alignments for each individual link in a parallel bus configuration regardless of the skew of the several transmission lanes.

Keeping in mind that for data neither the first four bits nor the last four bits of a 6B vector can be identical, the 5B/6B-T and the 10B/12B codes can easily be evaluated with the help of the trellis diagram of FIG. 22 and its characteristics can be summarized as follows:

1. The maximum run length (RL) is five with at most two contiguous runs of five in data mode, e.g, +011**00+000111**+110001+. Three contiguous runs of five are generated if the first two 6B vectors of the above example are followed by the comma character or certain other control characters: +011**00+000111+110000**-010011-.

2. For start-up purposes, the coding circuitry can generate a steady stream of runs of six, i.e. a symmetrical waveform at one twelfth the baud rate. The waveform "-110000'001111-110000'001111-..." is obtained by holding the encoder input at a steady K3 value (00111 K=1). The waveform "-1111100'000011-111100'000011-..." is obtained by holding the encoder input at a steady K15 (11110 K=1) value (refer to FIG. 23). These waveforms can be used to adjust the receiver circuits on parallel lanes to deal with skew and/or to find the 6B boundaries.

3. The minimum sustainable average transition density is three per 12-bit coded data interval, for example, the bit pattern '+100001-111000-011110+000111+' can be repeated indefinitely. Any single 12-bit interval may have as few as two transitions.

4. The code is dc-balanced with a maximum digital sum variation or running disparity variation of ±3. In the steady-state condition (ignoring irrelevant start-up abnormalities), all 6B vectors start and end with a running disparity of ±1.

5. The normalized maximum dc-offset is (13/6)=2.17 versus 1.9 for the 8B/10B-T code. The normalized maximum dc-offset is derived from the trellis diagram and is defined as the average area (bit interval x disparity) per bit-interval enclosed between the zero-disparity line and the outermost contour any valid code vector can traverse, which is '+110100+' or '-001011-' for the 5B/6B-T code. It is a better indicator for the low frequency behavior than the traditional DSV parameter. By simulations, it can be shown that at a 2Gbaud rate, a 3dB low frequency cutoff at 3.9MHz (0.195% of Baud rate) produces an eye amplitude closure of 0.5dB. A cutoff at 7.9MHz generates a closure penalty of 1dB for a worst case pattern. It is generally desirable to operate with a high low frequency cut-off in order to filter out low frequency noise from several sources and to permit small reactance values for ac-coupling. As an example, optical receiver designs usually have at least one high-pass filter on chip and so it is very important to reduce the size of the required capacitance. For equivalent results, the low frequency cut-off for the 5B/6B-T code must be set about 5.5% lower than for a conventional 8B/10B or the 8B/10B-T code described herein.

6. The maximum error spread caused by an error in the coded data is five contiguous decoded bits.

1. Coding Constraints and 12B Control Characters

A total of fifty-seven 12B control characters can be defined if needed without violating the existing run length and disparity constraints. A first set of 28 control characters is identified by K3 (110000 or 001111) in the first half segment, followed by the set of 15 balanced vectors which do not generate a run of six across the center division line. All these balanced 6B vectors are made disparity dependent as shown in the table shown in FIG. 23. Thirteen unbalanced 6B vectors also meet the constraints. Beyond that, one can define a new special 6B vector K15 with a leading run of four as listed in the table shown in FIG. 23. This vector is restricted to the second half of the 12B characters. So K3 followed by K15 yields another 12B control character which can be recognized as such by the presence of a special vector in either half-segment. A second set of 28 control characters is identified by the presence of K15 in the second half segment and the first half segment restricted to vectors which do not generate a run of six across the center boundary. Essentially, this second set is the first set of 12B characters with reversed bit order in the coded domain. Interestingly, there is also a second comma character with the same alignment with respect to the 12B boundaries, but the comma sequence is in reverse order ('110010000011' or '001101111100'). The definitions for the second set of 28 control characters are not included in the table shown in FIG. 23.

2. Comma Sequence for the 10B/12B Code

The preferred comma patterns is '11111011' or its complement '00000100'. The complete 12-bit comma character is "+110000 010011-" or "-001111 101100+" and it is generated by K3.K2 with some coder circuit modification to complement K2 in special characters when the starting disparity is positive. In the absence of errors, it is sufficient to monitor the seven bold digits for comma detection. An alternate comma pattern is the above bit sequence in reverse order "+110010 000011-" or "-001101 111100+".

Once vector alignment at the receiver has been established, both the 5B/6B-T and the 10B/12B code operate identically for data. The only differences are related to control characters and the comma sequence. Of course, alignment can be established by means other than a comma, e.g., a check for coding violations and step by step changes in alignment, or with the sequence of runs of six as described in list item 2 just above.

3. Implementation

In the circuit example described here, a limited set of fourteen 12B vectors is implemented to keep the circuits simple. The table shown in FIG. 24 lists the 6B vectors which may follow contiguously the special character K3 to form a 12B control character. With this limited set, only K2 and K3 are handled differently from their normal data mode D2 and D3, respectively, for encoding and decoding. K2 is a balanced vector made disparity dependent to form a comma sequence together with the preceding K3 character regardless of the running disparity at the start of K3. K3 has already been described as part of the 8B/10B-T code.

4. 5B/6B-T Encoding for 10B/12B Format

An encoding portion of a 10B/12B encoder is shown in FIG. 25A, while FIG. 25B shows a disparity control portion of the 10B/12B encoder. The AO block I152 and AOI blocks I348, I353, I280, and I354 are AND-OR and AND-OR-INVERT blocks, respectively, as described above in reference to FIGS. 9A and 9B. The circuit shown in FIGS. 25A and 25B takes a five-bit source vector defined by PA, PB, PC, PD, and PE and creates a six-bit coded vector defined by PCa, PCb, PCc, PCd, PCe, and PCi, according to the trellis diagram of FIG. 22 and the tables shown in FIGS. 14 and 24. The signal PCMPLS6 comes from the disparity control circuit shown in FIG. 25B. This signal is used to invert the primary coded vector abcdei to create the alternate coded vector (abcdei)'. This inversion is performed according to disparity rules.

The bit encoding for K2 is unchanged from data mode. However, the equation for the encoding for bit "i" is slightly changed because the value of K attached to the uncoded vector value '01000' may now be a one which would prevent the i-bit from being forced to one. The added term "$+A'$" is shown in brackets in the revised equation below:

$$i = C' \cdot D' \cdot E' \cdot (K'\{+A'\}) + A' \cdot B' \cdot C' \cdot D' + A' \cdot B' \cdot E' \cdot (C \neq D) + A \cdot B \cdot C \cdot D$$

This equation is equivalent to:

$$i = C' \cdot D' \cdot E' \cdot (K \cdot A)' + A' \cdot B' \cdot C' \cdot D' + A' \cdot B' \cdot E' \cdot (C \neq D) + A \cdot B \cdot C \cdot D$$

5. Disparity Control for 10B/12B Format

Because of the expanded set of 6B control vectors, the disparity controls have to be modified and some circuit simplifications are not applicable. In the equation for PDRS6, the last term $E' \cdot K$ must be expanded to $K \cdot A \cdot B \cdot C' \cdot D' \cdot E'$ to identify the K2 vector.

$$PDRS6 = E' \cdot (A \neq B) \cdot (C \neq D) + A' \cdot (B \neq C) \cdot (D \neq E)$$
$$+ B' \cdot C' \cdot E \cdot (A \neq D) + K \cdot A \cdot B \cdot C' \cdot D' \cdot E'$$

The vector K2 is an addition to the set of vectors which require a negative entry disparity. Within the limited set of the table shown in FIG. 24, it can be identified by the term $A' \cdot C' \cdot E' \cdot K$, surrounded by brackets, and so the equation for the required negative entry disparity becomes:

$$NDRS6 = A \cdot B \cdot C \cdot D' + C \cdot D \cdot E \cdot (A \neq B) + A \cdot B \cdot C' \cdot D \cdot E + \{A' \cdot C' \cdot E' \cdot K\}$$

The vector K2 is also an addition to the set of balanced S6 vectors:

$$BALS6 = (D \neq E) \cdot (A \oplus B \oplus C)' + D \cdot E \cdot (A \oplus B \oplus C)$$
$$+ A' \cdot B' \cdot D' \cdot E' + C' \cdot D' \cdot E' \cdot K'$$
$$+ A \cdot B \cdot C \cdot E' + \{A' \cdot C' \cdot E' \cdot K\}$$

6. 6B/5B-T Decoding of 12B Format

FIGS. 26A and 26B show a 12B/10B bit decoding circuit and a disparity and error checking circuit, respectively, of the 12B/10B decoding circuit. The bit decoding circuit of FIG. 26A takes a six-bit coded vector defined by PCa, PCb, PCc, PCd, PCe, and PCi and produces a five-bit source vector defined by PA, PB, PC, PD, and PE. The AO blocks I495, I496, I504, I503, and I448 and AOI blocks I311 and I367 are AND-OR and AND-OR-INVERT blocks, respectively, as described above in reference to FIGS. 9A and 9B.

The alternate vector for K2 "101100" is normally decoded to D13 = **1*0*1*1***0 in the B6 format, but if the preceding 6B vector is K3, then it is decoded to D2 = 01000, i.e., the first four bits are complemented. Since the alternate 6B vector for D/K18 "101101" is decoded to D18 = **0*1*0*0*1***, i.e., the first four bits are also complemented, one can specify that the first four bits are complemented if $(K3LAST \cdot a \cdot b' \cdot c \cdot d \cdot e')$ is true. The "i" bit can be ignored. However, because the full logic term including i' is required for the function PDRR6 below, the implementation does not make use of the possible simplification for applications which require disparity checks.

Therefore:

$$CMPLa = CMPL5 + b' \cdot i \cdot (a \neq c) \cdot (d \neq e)\{+ K3LAST \cdot a \cdot b' \cdot c \cdot d \cdot e' \cdot i'\}$$
$$CMPLb = CMPL5 + a' \cdot b' \cdot e' \cdot i \cdot (c \neq d) + a' \cdot b' \cdot c \cdot i \cdot (d \neq e)$$
$$\{+ K3LAST \cdot a \cdot b' \cdot c \cdot d \cdot e' \cdot i'\}$$

$$CMPLc = CMPL5 + a \cdot b' \cdot c \cdot d' \cdot e' \cdot i \{+ K3LAST \cdot a \cdot b' \cdot c \cdot d \cdot e' \cdot i' \}$$

$$CMPLd = CMPL5 +$$

$$b' \cdot i \cdot (a \neq c) \cdot (c \neq d) \cdot (d \neq e) \{+ K3LAST \cdot a \cdot b' \cdot c \cdot d \cdot e' \cdot i' \}$$

In the circuit diagram shown in FIG. 26A, the net name n30 stands for $K3LAST \cdot a \cdot b' \cdot c \cdot d \cdot e' \cdot i'$, and $n0 = a \cdot b \cdot i \cdot (c + d + e) + c \cdot i \cdot (a + b) \cdot (d + e)$.

7. Disparity Checks

For the limited set of control characters implemented, the primary vector K2 = 010011 and the alternate vector K2 = 101100 require a negative and positive entry disparity, respectively, when following K3. So the positive required front end disparity PDRR6 is augmented by the term:

$$K3LAST \cdot a \cdot b' \cdot c \cdot d \cdot e' \cdot i' = n30$$

The equation for the positive required front end disparity PDRR6 must include n30:

$$PDRR6 = a' \cdot b' \cdot c' + d' \cdot e' \cdot i' \cdot (a \cdot b \cdot c)'$$

$$+ (a' \cdot b' + a' \cdot c' + b' \cdot c') \cdot (d' \cdot e + d' \cdot i' + e' \cdot i') + n30$$

The negative required front end disparity NDRR6 is augmented by the term:

$$K3LAST \cdot a' \cdot b \cdot c' \cdot d' \cdot e \cdot i = n33$$

The equation for the negative required front end disparity NDRR6 must include n33:

$$NDRR6 = a \cdot b \cdot c + d \cdot e \cdot i \cdot (a + b + c)$$

$$+ (a \cdot b + a \cdot c + b \cdot c) \cdot (d \cdot e + d \cdot i + e \cdot i) + n33$$

More validity checks could be implemented in the 12B domain, especially with regard to invalid control character combinations. It is expected that most applications will not need those checks since most applications are expected to be focused on 5B/6B-T code and will use the 12B format only for start-up and special conditions. In other applications with multiple parallel lanes with error correction, the disparity checks at the receiver can be dispensed of because such errors are detected by other means.

8. Pipe-Lined Versions of Encoder and Decoder Circuits

Staggering the timing for the 6B and 4B parts with a few additional circuits to relieve timing and latency problems has been described above. If that simple solution is not enough, the individual circuits may have to be executed in two (or more) steps which adds latency and a modest amount of extra circuits.

For some applications such as very high speed parallel busses, a pipe-lined structure is preferable over several parallel CoDecs multiplexed into a line to overcome circuit delay problems. In this context, pipe-lining involves taking two basic clock intervals to obtain the coded or decoded results by inserting latches at suitable nodes. In the circuit diagram of the encoder shown in FIG. 9A, the six signals Nba through Pi at the upper center might by stored in a latch and the Exclusive OR function is then performed in the next clock cycle. To best exploit this technique, all circuits should be reevaluated. In the example cited, the signal path to Pi is longer than the encoding circuits for the other five bits because of the AO2222 gate which can be decomposed into two parallel AOI22 gates which are then combined by a NOR2 gate in the next cycle at the cost of an extra latch. In the lower right quadrant of the diagram, the signals PS1aGH, PS2aGH, PFGHKy, NF, and PGnHn can be latched for comparable circuit delays in the two cycles.

For another example of circuit changes in the pipe-lined version refer to the circuit shown in FIG. 9B in the upper right quadrant. The signals Pn1, Pn2, and Pn3 are all gated by a common input PDFS6 to eliminate a gating level. In the pipe-lined version with less critical timing, this gating function might be applied to the signal Pn1on2on3 instead, which then provides a larger margin for the signal PDFS6.

Generally, the pipe-lined version also provides more freedom to choose signal polarities for minimum delay since an inversion can easily be executed as part of the latch function with a lesser penalty than with the insertion of inverters. Sometimes this may also help to eliminate a gating level otherwise dictated by polarity considerations.

It is to be understood that the embodiments and variations shown and described herein are merely illustrative of the principles of this invention and that various modifications may be implemented by those skilled in the art without departing from the scope and spirit of the invention.

5